$O(1)$ Insertion for Random Walk d-ary Cuckoo Hashing up to the Load Threshold

Tolson Bell* and Alan Frieze†
Department of Mathematical Sciences
Carnegie Mellon University
Pittsburgh, PA 15213
U.S.A.

January 25, 2024

Abstract

The random walk d-ary cuckoo hashing algorithm was defined by Fotakis, Pagh, Sanders, and Spirakis to generalize and improve upon the standard cuckoo hashing algorithm of Pagh and Rodler. Random walk d-ary cuckoo hashing has low space overhead, guaranteed fast access, and fast in practice insertion time. In this paper, we give a theoretical insertion time bound for this algorithm. More precisely, for every $d \geq 3$ hashes, let c_d^* be the sharp threshold for the load factor at which a valid assignment of cm objects to a hash table of size m likely exists. We show that for any $d \geq 4$ hashes and load factor $c < c_d^*$, the expectation of the random walk insertion time is $O(1)$, that is, a constant depending only on d and c but not m.

1 Introduction

1.1 Random Walk d-ary Cuckoo Hashing

In random walk d-ary cuckoo hashing, the goal is to store objects X in a hash table Y given d hash functions $f_1, \dots, f_d : X \to Y$. Following previous literature, we will take each hash function to be a uniformly random function from X to Y. When a new object x is inserted, a random $i \in [d]$ is chosen, and x is placed into position $h_i(x)$. If $h_i(x)$ was already occupied, we remove its previous occupant, x_2, and reinsert x_2 by the same algorithm (choosing a new $i_2 \in [d]$ and putting x_2 into $h_{i_2}(x_2)$). This iterative algorithm terminates when we insert an object into an empty slot.

An object x is accessed by checking $h_1(x), \dots, h_d(x)$, which takes constant time for constant d. If we want to remove x, we simply delete it from its slot in the hash table. Thus access and deletion are both guaranteed to be fast.

Let $n = |X|$ and $m = |Y|$. We represent the hash functions as a bipartite graph with vertex set (X, Y), and for each $x \in X$, edges from x to $h_1(x), \dots, h_d(x)$. For a set $W \subseteq X$, we

*thbell@cmu.edu. Research supported in part by NSF Graduate Research Fellowship grant DGE 2140739.
†frieze@cmu.edu. Research supported in part by NSF grant DMS 1952285

let $N(W)$ denotes its set of neighbors in Y. An analogous definition is assumed for $Z \subseteq Y$. Finally, we replace $N(\{u\})$ by $N(u)$ for singleton sets.

For this insertion to terminate, it must be true that there is an assignment of every object to a slot such that no slot has more than one object and every object x is assigned to $h_i(x)$ for some $1 \leq i \leq d$. This can be represented as a matching of size n in the bipartite graph. We know by Hall's Theorem that such a matching exists if and only if $|N(W)| \geq |W|$ for every $W \subseteq X$.

All asymptotics in this paper are written for $m, n \to \infty$ as $n = cm$ for fixed $d \in \mathbb{N}$ and fixed load factor $c \in (0, 1)$. There is a sharp threshold c_d^*, called the load threshold, for a matching of size n to exist in the bipartite graph; that is, there is a constant c_d^* such that as $n, m \to \infty$ with $n = cm$, if $c < c_d^*$ then the probability of such a matching goes to 1, and if $c > c_d^*$ then the probability of such a matching goes to 0. Our result is the following:

Theorem 1.1. *Assume that we have $d \geq 4$, $c < c_d^*$, and $n = cm$. Then with high probability over the random hash functions, we have that the expected insertion time for the random walk insertion process is $O(1)$.*
Additionally, under the same conditions, there is a constant $C = \Theta(1)$ such that for sufficiently high n and all $\ell \in \mathbb{N}$, the probability of the random walk taking more than ℓ steps is at most $Ce^{-\ell^{.009}}$.

In other words, our main result is that the expected insertion time is a constant depending only on d and c but not n or m. We did not try to optimize the constant. Our result is also true beginning with any arbitrary assignment of objects to slots in the hash table.

Note that we are required to take our statement to only hold with high probability over the choices of hash functions, as there is a non-zero chance that the hash functions will not have any valid assignment of objects to slots (will fail Hall's condition) and thus will have infinite insertion time.

The second part of Theorem 1.1 gives super-polynomial tail bounds on the insertion time. The exponent 0.009 can be made to tend towards 1 as $d \to \infty$.

1.2 Applications and Relation to Previous Literature

Standard cuckoo hashing was invented by Pagh and Rodler in 2001 [23], and has been widely used in both theory and practice. Their formulation, though originally phrased with two hash tables, is essentially equivalent to the case $d = 2$ of the algorithm described here. They showed that for all $c < c_2^* = 0.5$, one can get $O(1)$ expected insertion time, an analysis that was extended by Devroye and Morin [4, 23].

d-ary cuckoo hashing was invented by Fotakis, Pagh, Sanders, and Spirakis in 2003 [9]. The main advantage of increasing d above 2 is that the load threshold increases. Even going from $d = 2$ to $d = 3$, the threshold c_d^* goes from 0.5 to ≈ 0.918, that is, with just one more hash function, we can utilize 91% of the hash table instead of 49%. The corresponding tradeoff is that the access time increases linearly with d. d-ary cuckoo hashing, also called generalized cuckoo hashing or improved cuckoo hashing, "has been widely used in real-world applications" [24].

The exact value for c_d^* for all $d \geq 3$ was discovered in 2009 via independent works by a number of authors [5, 11, 15]. This combinatorial problem of finding the matching threshold in these random bipartite graphs (which can also be viewed as d-uniform hypergraphs) is directly related to other problems like d-XORSAT [5] and load balancing [10, 17].

The primary insertion algorithm analyzed by Fotakis, Pagh, Sanders, and Spirakis was not random walk insertion, but rather was BFS insertion. In BFS insertion, instead of selecting a random $i \in [d]$ and hashing x to $h_i(x)$, the algorithm finds the insertion path minimizing the number of rehashes, that is, chooses the "shortest" insertion path instead of a random one. That is, $i_1, \ldots, i_\ell \in [d]$ are chosen such that ℓ is minimized, where x is to be hashed to $h_{i_1}(x)$, the removed object x_2 is to be hashed to $h_{i_2}(x_2)$, and so on until $h_{i_\ell}(x_\ell)$ is an empty slot. While BFS insertion requires more overhead to compute, it is easier to analyze theoretically than random walk insertion. Fotakis, Pagh, Sanders, and Spirakis proved that BFS insertion is $O(1)$ for load factor c when $d \geq 5 + 3\log(1/c)$ [9]. Though not explicitly stated, the results of Fountoulakis, Panagiotou, and Steger imply that this result extends to all $d \geq 3$ and $c < c_d^*$ [12].

Fotakis, Pagh, Sanders, and Spirakis also introduced the insertion algorithm we study, random walk insertion, describing it as "a variant that looks promising in practice", since they did not theoretically bound its insertion time but saw from experiments that its insertion time was fast [9]. Random walk insertion requires no extra space overhead or precomputation: very important for the use case of d-ary cuckoo hashing, situations requiring high load factor. In a 2009 survey on cuckoo hashing, Mitzenmacher raised the importance of proving theoretical bounds for random walk insertion, calling random walk insertion "much more amenable to practical implementation" and "usually much faster" than BFS insertion [21]. Other insertion algorithms than random walk or BFS have been proposed, which have provable $O(1)$ insertion [18] or more evenly distributed memory usage [7] while having lower memory overhead than BFS insertion. However, random walk insertion "is currently the state-of-art method and so far considered to be the fastest algorithm" [18].

For load factors somewhat below the load threshold and $d \geq 8$, the random walk insertion time was proven to be polylogarithmic by Frieze, Melsted and Mitzenmacher in 2009 [14]. Fountoulakis, Panagiotou, and Steger then were able to show polylogarithmic insertion time for all $d \geq 3$ and $c < c_d^*$. The exponent of their logarithm was anything greater than $1 + b_d$, where $b_d = \frac{d+\log(d-1)}{(d-1)\log(d-1)}$ [12]. Our proof uses some techniques and lemmas of these two papers.

The first $O(1)$ random walk insertion bound was proven by Frieze and Johansson, who showed that for any load factor c, there exists some d such that there is $O(1)$ insertion time for d hashes at load factor c [13]. However, their bounds only hold for large d and load factors significantly less than the load threshold, specifically, $c = 1 - O(\log(d)/d)$, while we know that $c_d^* = 1 - e^{-d} - o(e^{-d})$.

For lower d, Walzer used entirely different techniques to prove $O(1)$ random walk insertion up to the "peeling threshold". The strongest result here is in the case $d = 3$, where Walzer gets $O(1)$ insertion up to load factor $c = .818$, compared to the optimal value $c_3^* = .918$. Walzer pointed out that there was no $d \geq 3$ for which $O(1)$ insertion was known up to the load threshold, saying, "Given the widespread use of cuckoo hashing to implement compact dictionaries and Bloom filter alternatives, closing this gap is an important open problem for theoreticians" [25].

Theorem 1.1 is the first result to get $O(1)$ insertion up to the load threshold for any $d \geq 3$, and works for all $d \geq 4$. The state of the art results are summarized in the tables below:

d	c_d^*	Maximal load factor for $O(1)$ insertion	Insertion time at $c = (1-\epsilon)c_d^*$
2	[1]0.5	[1]0.5	[1]$O(1)$
3	[2]0.918	[3]0.818	[4]$O(\log^{3.664}(n))$
4	[2]0.977	[3]0.772	[4]$O(\log^{2.547}(n))$
5	[2]0.992	[3]0.702	[4]$O(\log^{2.152}(n))$
6	[2]0.997	[3]0.637	[4]$O(\log^{1.946}(n))$
7	[2]0.999	[3]0.582	[4]$O(\log^{1.818}(n))$
Large	[2]$1 - e^{-d} - o(e^{-d})$	[5]$1 - O(\frac{\log d}{d})$	[4]$O(\log^{1+(\log d)^{-1}+O(1/d)}(n))$

Bounds from prior work: [1][23, 4] [2][5, 11, 15] [3][25] [4][12] [5][13]

d	c_d^*	Maximal load factor for $O(1)$ insertion	Insertion time at $c = (1-\epsilon)c_d^*$
2	[1]0.5	[1]0.5	[1]$O(1)$
3	[2]0.918	[3]0.818	[7]$O(\log^{2.509}(n))$
4	[2]0.977	[6]0.977	[6]$O(1)$
5	[2]0.992	[6]0.992	[6]$O(1)$
6	[2]0.997	[6]0.997	[6]$O(1)$
7	[2]0.999	[6]0.999	[6]$O(1)$
Large	[2]$1 - e^{-d} - o(e^{-d})$	[6]$1 - e^{-d} - o(e^{-d})$	[6]$O(1)$

Bounds after our work: [6]Theorem 1.1 and [7]Also given in our proof

1.3 Future Work

The central open question is to remove the restriction $d \geq 4$ from Theorem 1.1, that is, to get $O(1)$ insertion up to the load threshold for $d = 3$. We are hopeful that the techniques in our paper can be extended to finish this final case.

The super-polynomial tail bounds on the insertion time in Theorem 1.1 can be made to tend towards being exponential tail bounds as $d \to \infty$. It would be interesting to show exponential tail bounds, as well as $O(1)$ insertion, for all $d \geq 3$.

It would also be interesting to give a stronger bound on the $o(1)$ term in our "with high probability" statements. A careful analysis of our and previous works ([11, 12]) shows that this probability could currently be taken to be $O(n^{-\beta})$ for some small $\beta = \Theta(1)$. By a union bound, the failure probability also implies that the $O(1)$ expected insertion time is robust to $O(n^{\beta})$ non-hash-dependent deletions and insertions, as long as the load factor stays below c.

Now that we have an insertion time independent of n, another avenue for future study is to optimize the insertion time in terms of d, c, and absolute constants.

It has been shown under some previous models of cuckoo hashing that the assumption of uniformly random hash functions can be relaxed to families of efficiently computable hash functions while retaining the theoretical insertion time guarantees [1, 3]. As our proof relies on similar "expansion-like" properties of the bipartite graph to previous work, we believe that Theorem 1.1 should still hold under practically computable hash families.

A different model for generalizing cuckoo hashing, proposed in 2007, gives a capacity greater than one to each hash table slot (element of Y), instead of (or in addition to) additional hash functions [6]. The load thresholds for this model are known for both two hashes [2, 8] and $d \geq 3$ hashes [10]. As in our model, $O(1)$ expected time for random walk insertion has been shown for some values below the load threshold [16, 25], but it remains open for any capacities greater than one to prove $O(1)$ insertion up to the load thresholds.

In general, it would be nice to extend our random walk insertion time guarantees to other modifications of cuckoo hashing, such as those that get good load factors with somewhat fewer hashes [26] or those that deal with the situation where a valid matching fails to exist [19, 20].

2 Determining the "Bad" Sets

Our techniques to prove Theorem 1.1 build off the techniques of Fountoulakis, Panagiotou, and Steger [12], who showed expansion-like properties of the bipartite hashing graph that hold with high probability. The main new ingredient is the introduction of specifically defined "bad" sets $B_1 \supseteq B_2 \supseteq \ldots$. In this section, we will give the definition of these bad sets and explain the overall proof structure. The main idea is that these B_i are complements of good sets G_i, and we will show in this section that a random walk starting from a vertex in G_i has at least a specified probability of finishing (reaching an unoccupied slot) in the next $O(i)$ steps.

Our most technical section, Section 5, shows that the size of these B_i decline exponentially in i. In Section 3, we will show that reaching a small set is unlikely, and thus the probability of a random walk reaching B_i declines sufficiently rapidly with i. In Section 4, we will explain how our proof also gives the second part of Theorem 1.1, which bounds the tail probabilities, not just the expectation, of the insertion time.

2.1 The Matching and BFS Distance

We will study the form of the random walk where at each object removal, we choose a random one of the $(d-1)$ other hashes for the object that was just kicked out (not returning it to the spot it was just kicked out of). Proving the expected run time of this is $O(1)$ also proves the same of the run time of choosing a random one of the d hashes each time (including the one it was just kicked from), as this just adds a delay of twice a $Geom((d-1)/d)$ random variable at each step in the previous random walk, which then just multiplies the expectation by $2/(d-1)$.

We will only consider the insertion of one element into the hash table. As the only "with high probability" statements in our proof are about the structure of the bipartite graph, this implies $O(n)$ time with high probability to build the hash table of n elements online. Let $\mathcal{M}_t$ be the assignment/matching of objects to slots at time t in the random walk. That is, $\mathcal{M}_t$ is a matching of size $n-1$, with the one currently swapping element unassigned, and $\mathcal{M}_0$ is the starting matching of size $n-1$ just before we insert the nth element. Let $U \subseteq Y$

be the set of open spots in the hash table, which stays the same at each time step while the algorithm is running (as the algorithm terminates when it hits an open slot).

Our proof only relies on expansion-like properties of the bipartite graph on (X, Y) that hold with high probability. In particular, given the random bipartite graph, our result holds for any arbitrary starting matching $\mathcal{M}_0$ of objects to slots.

Let $\underline{W_i^{(t)}(x)} \subseteq X$ be the set of all possible endpoints of a walk of length at most i starting from x and the matching $\mathcal{M}_t$ (so $|W_i^{(t)}(x)| \leq \sum_{j=0}^{i}(d-1)^j \leq (d-1)^{i+1}$).

The BFS distance of an object w from an object x under $\mathcal{M}_t$ is the minimal i such that $w \in W_i^{(t)}(x)$. We can define BFS distances involving sets in the natural way, by minimizing over elements of those sets. We can similarly define the BFS distance of a slot y from an object x as 1 plus the BFS distance between x and $N(y)$.

For every $u \in U$ that is at BFS distance j from x for some $j \leq i$, count $u_1, \ldots, u_{(d-1)^{i-j}}$ as distinct elements of $W_i^{(t)}(x)$. Let $\mathcal{U}_i^{(t)}$ then be the multiset of the $u \in U \cap W_i^{(t)}(x)$ with these multiplicities. We also have $W_0^{(t)}(x) = \{x\}$ and $x \in W_i^{(t)}(x)\ \forall\ i \in \mathbb{N}$. For $S \subseteq X$, we can similarly define $W_i^{(t)}(S) = \cup_{x \in S} W_i^{(t)}(x)$.

Lemma 2.1 (Corollary 2.3 of [12])**.** *Assume $n = cm$ for $c < c_d^*$. Then with high probability, we have that for any matching $\mathcal{M}$ and any $\alpha = \Theta(1) > 0$, there exists $M = \Theta(1)$ such that for the unoccupied vertices U of Y, we have that at most αn of the vertices of X are at BFS distance $> M$ from U.*

(Lemma 2.1 had initially been proven by the inventors of d-ary cuckoo hashing under the weaker condition $d \geq 5 + 3\log(c/(1-c))$ for $n = cm$ [9]. Note that all logarithms in our paper are natural.)

Let $\alpha > 0$ be sufficiently small (but still $\Theta(1)$, to be set later) and take the corresponding $M = \Theta(1)$ as in Lemma 2.1. For any $\mathcal{M}_t$, let $G^{(t)}$ be all vertices of X of BFS distance at most M from U. Whenever we are at a vertex $g \in G^{(t)}$, we have at least a d^{-M} chance that our random walk will finish in at most M more steps. (That is, there is at least a d^{-M} chance that our random walk will be the BFS path.) Therefore, the expected length on a random walk that stays inside $G^{(t)}$ at every time t is at most $d^M + M = \Theta(1)$. This shows intuitively that it suffices to only focus on the "worst" αn vertices for some $\alpha = \Theta(1) > 0$.

2.2 Definition of $B_i^{(t)}$

We will split up the bad set $X \setminus G^{(t)}$ into further worse and worse subsets defined based on $G^{(t)}$.

From any $x \in X$, there are $(d-1)^i$ equally likely walks of length $i \in \mathbb{N}$, given that we have $\mathcal{U}_i^{(t)}$ with proper multiplicities. Note that we are referring to a walk of length i throughout to refer to i reassignments, but really this is a walk of length $2i$ in the bipartite graph (X, Y).

Take $C_0 = \Theta(1)$ to be fixed later. For any $i \in \mathbb{N}$, we define

$$G_i^{(t)} = \left\{ x \in X : |W_i^{(t)}(x) \cap (G^{(t)} \cup \mathcal{U}_i^{(t)})| \geq \frac{(d-1)^i}{C_0 i^{.99}} \right\}$$

(We define $G_0^{(t)} = G^{(t)}$.) The definition of $G_i^{(t)}$ is useful for the following reason: if we have a random walk starting at some $x \in G_i^{(t)}$, we have at least a $(C_0)^{-1} i^{-.99}$ chance that the random walk will be in $G^{(t+j)}$ after some $j \leq i$ steps, as for each $w \in W_i^{(t)}(x)$, there is at least a $(d-1)^{-j} \geq (d-1)^{-i}$ chance we are at w after $j \leq i$ steps. (And note that $G^{(t)} = G^{(t+j)}$ for all $j \leq i$ while we have not yet reached $G^{(t)}$, as the BFS paths within $G^{(t)}$ remain unchanged.)

Therefore, for a random walk starting at some $x \in G_i^{(t)}$, we have at least a $d^{-M}C_0^{-1}i^{-.99}$ chance that the random walk will finish in at most $i + M$ steps, by reaching $G^{(t+j)}$ in $j \leq i$ steps and then taking the BFS path from there. This shows that the expected length of a random walk that stays within $\cup_{j=0}^{i} G_j^{(t)}$ at each time t is at most $C_0 d^M i^{.99} + i + d$: at each step, we are in some $G_j^{(t)}$, and thus by the previous paragraph have at least a $d^{-M}(C_0)^{-1}i^{-.99}$ chance of finishing in at most $j + M \leq i + M$ further steps.

Now, define our bad sets to be the complement of these,

$$B_i^{(t)} = X \setminus (\cup_{j=0}^{i} G_j^{(t)}).$$

So then we have $X = B_{-1}^{(t)} \supseteq B_0^{(t)} \supseteq B_1^{(t)} \supseteq \ldots$. By "reaching $B_i^{(t)}$" we mean that at some time step t we are in $B_i^{(t)}$. Then

$$\begin{aligned}
\mathbb{E}(|\mathrm{RW}|) &= \sum_{i=0}^{\infty} \mathbb{E}(|\mathrm{RW}| : \text{RW reaches } B_{i-1}^{(t)} \text{ but not } B_i^{(t)}) \mathbb{P}(\text{RW reaches } B_{i-1}^{(t)} \text{ but not } B_i^{(t)}) \\
&\leq \sum_{i=0}^{\infty} (C_0 d^M i^{.99} + i + M) \mathbb{P}(\text{RW reaches } B_{i-1}^{(t)} \text{ but not } B_i^{(t)}) \\
&\leq \sum_{i=0}^{\infty} (C_0 d^M i^{.99} + i + M) \mathbb{P}(\text{RW reaches } B_{i-1}^{(t)})
\end{aligned}$$

Thus, to prove that the expected length of the random walk is $O(1)$ as desired, it suffices to prove that

$$\exists\, C = O(1) \text{ such that } \mathbb{P}(\text{RW reaches } B_i^{(t)}) \leq C i^{-3} \text{ for all } i \in \mathbb{N}, \tag{1}$$

as then the above sum will be $O(\sum_{i=0}^{\infty} i^{-2}) = O(1)$. In fact, we will show the stronger bound that the probability of reaching $B_i^{(t)}$ decreases faster than any polynomial.

3 Probability of Reaching a Small Set

3.1 Neighbors of a Small Set

To show that reaching some bad set is unlikely, we want to upper bound the probability of reaching some small set. To do this, we need to bound the number of neighbors that a small set can have.

For $Z \subseteq X \cup Y$, let recall that $N(Z)$ refers to the neighbors of Z in the bipartite graph.

Lemma 3.1. *With high probability, there is not a set $Z \subseteq Y$ with $|Z| \leq n/12$ such that $|N(Z)| \geq 3d \log\left(\frac{n}{|Z|}\right) |Z|$.*

Proof. First, imagine fixing $Z \subseteq Y$ then randomly choosing the edges of our graph. Let $e(Z)$ be the number of edges incident to Z. Our bipartite graph has dn edges, and each has an independent $|Z|/m \le |Z|/n$ chance of landing in $|Z|$. Thus, we can assume $e(Z) \sim Bin(dn, |Z|/n)$ and $\mathbb{E}(e(Z)) = d|Z|$. By standard Chernoff bounds,

$$\mathbb{P}\left(e(Z) \ge 3d\log\left(\frac{n}{|Z|}\right)|Z|\right) \le \left(\frac{e}{3\log(n/|Z|)}\right)^{3d|Z|\log(n/|Z|)} \le e^{-3d|Z|\log(n/|Z|)} = \left(\frac{|Z|}{n}\right)^{3d|Z|}$$

Then

$$\begin{aligned}
&\mathbb{P}\left(\exists\ Z \subseteq Y \text{ s.t. } |N(Z)| \ge 3d\log\left(\frac{n}{|Z|}\right)|Z|\right)\\
&\le \sum_{i=1}^{n/12}\binom{m}{i}\left(\frac{i}{n}\right)^{3di} \le \sum_{i=1}^{n/12}\left(\frac{2en}{i}\right)^{i}\left(\frac{i}{n}\right)^{3di} = \sum_{i=1}^{n/12}\left(2e\left(\frac{i}{n}\right)^{3d-1}\right)^{i}\\
&\le \sum_{i=1}^{\log^2(n)} 2e\left(\frac{\log^2(n)}{n}\right)^2 + \sum_{i=\log^2(n)}^{n/12}\left(2e\left(\frac{1}{12}\right)^2\right)^{\log^2(n)} = o(1/n).
\end{aligned}$$

$\square$

Now, for $x \in X$ and $j \in \mathbb{N}$, let $W_{-j}^{(t+j)}(x) = \{w \in X : x \in W_j^{(t)}(w)\}$.

Lemma 3.2. *For any $S \subseteq X$, $|S| \le n/12$, and $t \in \mathbb{N}$, we have $|W_{-j}^{(t+j)}(S)| \le \left(3d\log\left(\frac{n}{|S|}\right)\right)^j |S|$.*

Proof. We can prove this inductively as a corollary of the lemma above. We see that it is true for $j = 0$. Then note that $W_{-j}^{(t+j)}(S) = W_{-1}^{(t+1)}(W_{-j+1}^{(t+j)}(S)) = N(Z)$ where $Z \subseteq Y$ is the spots occupied by $W_{-j+1}^{(t+j)}(S)$, which thus has the same cardinality of $W_{-j+1}^{(t+j)}(S)$.

So using Lemma 3.1 and recalling that $S \subseteq W_{-j+1}^{(t+j)}$, we have

$$\begin{aligned}
|W_{-j}^{(t+j)}(S)| \le 3d\log\left(\frac{n}{|W_{-j+1}^{(t+j)}(S)|}\right)|W_{-j+1}^{(t+j)}(S)| &\le 3d\log\left(\frac{n}{|S|}\right)|W_{-j+1}^{(t+j)}(S)|\\
&\le \left(3d\log\left(\frac{n}{|S|}\right)\right)^j |S|
\end{aligned}$$

as desired.

(Note that if we ever have $|W_{-j+1}^{(t+j)}(S)| \ge n/12$ (so Lemma 3.1 can't be applied), then we have $|W_{-j}^{(t+j)}(S)| \le 3d\log\left(\frac{n}{|S|}\right)|W_{-j+1}^{(t+j)}(S)|$ anyway, as the right side of the equation is then more than n.) $\square$

3.2 Probability of reaching $B_i^{(t)}$

In Section 5, we will prove that the $B_i^{(t)}$ have exponentially decreasing sizes, proving the following lemma:

Lemma 3.3. *With high probability over the choice of $d \geq 4$ hashes, there is a $C = \Theta(1)$ such that $|B_i^{(t)}| \leq Cn2^{-i}$ for any matching $\mathcal{M}_t$ and for all $i \in \mathbb{N}$.*

Because the proof of Lemma 3.3 is a bit more technical, we defer it to the end of our paper. We finish our proof by putting Lemma 3.2 and Lemma 3.3 together to show that $B_i^{(t)}$ is unlikely to be reached:

Lemma 3.4. *The probability that our random walk reaches $B_i^{(t)}$ is at most $C'i^{-3}$ for some constant C'.*

As discussed in Section 2, this proves $O(1)$ insertion time.

Proof. The previous two sections put together give us that for all $i, t \in \mathbb{N}$,

$$|W_{-t}(B_i^{(t)})| \leq \left(3d\log\left(\frac{n}{|B_i^{(t)}|}\right)\right)^t |B_i^{(t)}|$$
$$\leq C\left(3d\log\left(\frac{n}{C2^{-i}n}\right)\right)^t 2^{-i}n \leq C\left(3d(i-\log(C))\right)^t 2^{-i}n.$$

Then

$$\begin{aligned}
\mathbb{P}(B_i \text{ reached}) &\leq \sum_{t=0}^{C_0 d^M i^{.999}} \mathbb{P}(\text{step } t \text{ of the RW is in } B_i^{(t)}) \\
&\qquad + \mathbb{P}(B_i^{(t)} \text{ reached on step } t \text{ for some } t > C_0 d^M i^{.999}) \\
&\leq \sum_{t=0}^{C_0 d^M i^{.999}} \mathbb{P}(\text{RW starts in } W_{-t}^{(t)}(B_i^{(t)})) \\
&\qquad + \mathbb{P}(\text{RW reaches } B_i^{(t)} \mid \text{RW walks in } X \setminus B_i^{(t)} \text{ for } \geq C_0 d^M i^{.999} \text{ steps}) \\
&\leq \sum_{t=0}^{C_0 d^M i^{.999}} \frac{|W_{-t}^{(t)}(B_i^{(t)})|}{n} \\
&\qquad + \mathbb{P}(\text{RW reaches } B_i^{(t)} \mid \text{RW walks in } X \setminus B_i^{(t)} \text{ for } \geq C_0 d^M i^{.999} \text{ steps}) \\
&\leq \sum_{t=0}^{C_0 d^M i^{.999}} \frac{|W_{-t}^{(t)}(B_i^{(t)})|}{n} + (1-(C_0)^{-1}d^{-M}i^{-.99})^{C_0 d^M i^{.999}} \\
&\qquad (\text{as at each step in } X \setminus B_i^{(t)} \text{ there is a probability} \geq (C_0)^{-1}d^{-M}i^{-.99} \text{ of} \\
&\qquad \text{finishing in} \leq i+M \text{ further steps without ever hitting } B_i^{(t)}) \\
&\leq \sum_{t=0}^{C_0 d^M i^{.999}} \frac{|W_{-t}(B_i^{(t)})|}{n} + e^{-i^{.009}} \\
&\leq C_0 d^M i^{.999}\left(3d(i-\log(C))\right)^{C_0 d^M i^{.999}} 2^{-i} + e^{-i^{.009}} \\
&= O(1.5^{-i}) + e^{-i^{.009}} = O(e^{-i^{.009}}).
\end{aligned}$$

$\square$

This proves (1) and completes the proof of $O(1)$ insertion time (once Lemma 3.3 is proven in Section 5).

4 Concentration Bounds

Our proof also shows that the tail bounds on the insertion time concentrate super-polynomially:

Lemma 4.1. *Assume that we have $d \geq 4$, $c < c_d^*$, and $n = cm$. With high probability over the choice of hash functions, there is a constant $C = \Theta(1)$ such that for sufficiently high n and all $\ell \in \mathbb{N}$, the probability of the random walk taking more than ℓ steps is at most $Ce^{-\ell^{.009}}$.*

Proof. Take $i \in \mathbb{N}$. In order for the random walk to take at least $C_0 d^M i^{.99} + i + M$ steps, either we reach B_i in at most $C_0 d^M i^{.99}$ steps, or we walk outside of B_i for at least $C_0 d^M i^{.99}$ steps without choosing to finish in the next $i + M$ steps. Section 3 shows that the probability of the former is $O(1.5^{-i})$ and the probability of the latter is $O(e^{-i^{.009}})$.

Thus, taking $\ell = C_0 d^M i^{.99} + i + M$, the probability of the random walk lasting at least ℓ steps is $O(e^{-i^{.009}}) = O(e^{-\ell^{.009}})$, which is super-polynomially decreasing in ℓ. $\square$

In fact, tracing through our proof, we see that .009 could be any value less than $1 - b_d$ for a $b_d \leq \frac{(d-1)+\log(d-1)}{(d-1)\log(d-1)}$, and we have $b_d \to 0$ as $d \to \infty$. So in other words, the tail bounds tend towards being an exponential decrease as $d \to \infty$.

5 Bounding the sizes of $B_i^{(t)}$

The remaining task is to show that the sizes of the $B_i^{(t)}$ decline like $O(2^{-i})$. The results in this section rely heavily on results of Fountoulakis, Panagiotou, and Steger [12].

Recall that for any $\mathcal{M}_t$, we have $W_1^{(t)}(S) = \cup_{x \in S} W_1^{(t)}(x)$, in other words, the set of all $w \in X$ that we could reach by one cuckoo iteration starting somewhere in S (and including $S \subseteq W_1(S)$). We also must have $|W_1^{(t)}(S)| \leq |S| + (d-1)|S| = d|S|$. The following lemma shows that for small S, $|W_1^{(t)}(S)|$ is close to its upper bound.

Lemma 5.1 (Proposition 2.4 of [12])**.** *For any $1 \leq s \leq |X|/d$, define*

$$x_s = \begin{cases} 0 & \text{if } |S| \leq \log\log(n) \\ \frac{\log_d((d-1)e^d)}{\log(|X|/|S|)-1} & \text{if } \log\log(n) \leq |S| \leq |X|/d \end{cases}$$

With high probability, we have that for all $S \subseteq X$ with $|S| \leq |X|/d$ that

$$|N(S)| \geq (d - 1 - x_{|S|})|S|.$$

Lemma 5.1 is good enough for our proof to go through for $d \geq 6$. We defer the $d \leq 5$ cases to the computation-heavy Subsection 5.2, where we will prove a form of Lemma 5.1 with stronger parameters.

5.1 Bounding $|B_i^{(t)}|$ for $d \geq 6$

Let $a_d = (d-1)e^d$. Now, following [12], let $s_0 = 1$ and inductively set $s_i = (d-1-x_{s_{i-1}})s_{i-1}$. We cite another lemma from [12]:

Lemma 5.2 (Claim 4.5 of [12])**.** *For every $d \geq 3$ and $\gamma > 0$ there exists $\epsilon_0 = \epsilon_0(\gamma, d) = \Theta(1)$ such that for all $0 < \epsilon < \epsilon_0$ and n sufficiently large the following is true. Set*

$$T = \log_{d-1}(n) + \left(\frac{\log(a_d)}{(d-1)\log(d-1+\gamma)}\right)\log_{d-1}(\log_{d-1}(n)).$$

Then $s_T > \epsilon n$.

Take $\gamma = \Theta(1)$ sufficiently small such that $\frac{\log(a_d)}{(d-1)\log(d-1+\gamma)} < .98$ (noting that $\frac{\log(a_d)}{(d-1)\log(d-1)} < .98$ for $d \geq 6$) and take the corresponding $\epsilon_0 = \Theta(1)$. In Lemma 2.1, take $\alpha \leq \epsilon_0/(2(d-1)))$.

Now, clearly there is some R such that $2\alpha n \leq s_R \leq \epsilon_0 n$, as we multiply s_i by at most $d-1$ at each step. Fix some such R and note $\log_{d-1}(2\alpha n) \leq R \leq \log_{d-1}(n) + \left(\frac{\log(a_d)}{(d-1)\log(d-1+\gamma)}\right)\log_{d-1}(\log_{d-1}(n))$ by Lemma 5.2.

Lemma 5.3. *Under any matching $\mathcal{M}_t$, we have $B_R^{(t)} = \emptyset$.*

Proof. Assume there were some $x \in B_R^{(t)}$. Then we would inductively get that $|W_i^{(t)}(x)| \geq s_i$ (remembering to count the elements of U with proper multiplicities in $\mathcal{U}_i^{(t)}$), so in particular

$$|W_R^{(t)}(x)| \geq 2\alpha n \implies |W_R^{(t)}(x) \cap (G^{(t)} \cup \mathcal{U}_i^{(t)})| \geq \alpha n \implies$$
$$(d-1)^{-R}|W_R^{(t)}(x) \cap (G^{(t)} \cup \mathcal{U}_i^{(t)})| \geq \left(\frac{1}{n\log^{.98}(n)}\right)\alpha n > \frac{1}{C_0 R^{.99}}$$

when assuming $C_0 > \alpha^{-1}$. This, however, contradicts that we need by definition that $|W_R^{(t)}(x) \cap (G^{(t)} \cup \mathcal{U}_i^{(t)})| \leq \frac{(d-1)^R}{C_0 R^{.99}}$ for all $x \in B_R^{(t)}$. $\square$

So we have successfully shown that for $i \geq R$, $B_i^{(t)}$ is empty. To bound the sizes of lower $B_i^{(t)}$, we need to look closer at the proof of Lemma 5.2 and use some additional lemmas of [12].

Lemma 5.4 (Claim 4.4 of [12])**.** *Let $t \geq \log_{d-1}(\log(\log(n))) + 1$. For every $\epsilon > 0$ sufficiently small, if $s_t \leq \epsilon n$, then for all $0 \leq i \leq t - \log_{d-1}(\log(\log(n))) - 1$, we have*

$$x_{s_{t-i}} \leq \frac{\log_d(a_d)}{i\log_d(d-1-\gamma) + \log_d(1/\epsilon) - 1}$$

where $\gamma = \frac{\log_d(a_d)}{\log_d(1/\epsilon)-1}$.

Recall that we have set ϵ small enough such that $\frac{\log(a_d)}{(d-1)\log(d-1+\gamma)} < .98$.

Lemma 5.5 (Proposition 4.1 of [12])**.** *For any constants $\zeta, \eta > 0$ we have that whenever $D = D(\zeta, \eta)$ is sufficiently large then*

$$\prod_{k=1}^{i}\left(1 - \frac{\zeta}{k\eta + D}\right) \geq i^{-\zeta/\eta}(\eta D)^{-\zeta/\eta}e^{-\zeta^2/(\eta D)} \qquad \textit{for all } i \geq 2/\eta$$

The previous two lemmas combine to prove the following:

Lemma 5.6 ([12])**.** *For all* $1 \le i \le .99 \log_{d-1}(n)$*, we have* $s_{R-i} \le \frac{C_0 \alpha n}{2(d-1)^i} i^{.99}$.

Proof. This is proved following the first half of the proof of Claim 4.5 of [12].

Because $s_R \le \epsilon_0 n$, we can use Lemma 5.4 and the definition $s_i = (d-1-x_{s_{i-1}})s_{i-1}$ to get

$$s_{R-i} \le \frac{\epsilon n}{\prod_{k=1}^{i}(d-1-x_{s_{R-i}})} \le \frac{\epsilon_0 n}{(d-1)^i \prod_{k=1}^{i} \frac{\log_d(a_d)/(d-1)}{k \log_d(d-1-\gamma)+\log_d(1/\epsilon_0)-1}}$$

then using Lemma 5.5 with $\zeta = \frac{\log_d(a_d)}{d-1}$ and $\eta = \log_d(d-1-\gamma)$ we get for all $i \ge 4 \ge 2/\eta$ that

$$s_{R-i} \le \frac{\epsilon_0 n}{(d-1)^i \prod_{k=1}^{i} \frac{\log_d(a_d)/(d-1)}{k \log_d(d-1-\gamma)+\log_d(1/\epsilon_0)-1}} \le C_{\epsilon_0,d} \frac{\epsilon_0 n}{(d-1)^i} \left(i^{\left(\frac{\log_d(a_d)}{(d-1)\log_d(d-1-\gamma)}\right)} \right),$$

which is less than the desired quantity as long as we take $C_0 > 4(d-1)C_{\epsilon_0,d}$ (recalling $\alpha = \epsilon_0/(2(d-1))$ and we set γ such that $\frac{\log(a_d)}{(d-1)\log(d-1-\gamma)} < .98$). Assuming $C_0 > 4$ then also works for $1 \le i \le 3$. $\square$

Lemma 5.7. $|B_i^{(t)}| \le C_0 \frac{\alpha n}{(d-1)^i} i^{.99}$ *for any matching* $\mathcal{M}_t$ *and for all* $1 \le i \le .9 \log_{d-1}(n)$*.*

Proof. Note that we have for every $x \in B_i^{(t)}$ that $|W_i^{(t)}(x) \cap (G^{(t)} \cup \mathcal{U}_i^{(t)})| \le \frac{(d-1)^i}{C_0 i^{.99}}$, so we also know

$$|W_i^{(t)}(S) \cap (G^{(t)} \cup \mathcal{U}_i^{(t)})| \le |S| \frac{(d-1)^i}{C_0 i^{.99}} \text{ for any } S \subseteq B_i^{(t)}. \tag{2}$$

Assume for contradiction that we had $|B_i^{(t)}| > C_0 \frac{\alpha n}{(d-1)^i} i^{.99} \ge 2 s_{R-i}$. Then in particular, we could find a $S \subseteq B_i^{(t)}$ with $s_{R-i} \le |S| \le 2 s_{R-i}$. Then $|W_i^{(t)}(S)| \ge s_R \ge 2\alpha n$, so

$$|W_i^{(t)}(S) \cap (G^{(t)} \cup \mathcal{U}_i^{(t)})| \ge \alpha n = \left(C_0 \frac{\alpha n}{(d-1)^i} i^{.99} \right) \frac{(d-1)^i}{C_0 i^{.99}} \ge |S| \frac{(d-1)^i}{C_0 i^{.99}}$$

contradicting (2). $\square$

So we now know by Lemma 5.7 that $|B_i^{(t)}|$ declines exponentially for $2 \le i \le .9 \log_{d-1}(n)$, and we know by Lemma 5.3 that $B_i^{(t)} = 0$ for $i \ge \log_{d-1}(n) + \left(\frac{\log(a_d)}{(d-1)\log(d-1+\gamma)} \right) \log_{d-1}(\log_{d-1}(n))$. This, plus knowing that $|B_i^{(t)}|$ is monotone decreasing in i, gives us the result we want:

Lemma 3.3. *There is a* $C = \Theta(1)$ *such that* $|B_i^{(t)}| \le C n 2^{-i}$ *for any matching* $\mathcal{M}_t$ *and for all* $i \in \mathbb{N}$*.*

Proof. First, we can take $C_1 = \Theta(1)$ large enough such that

$$|B_i^{(t)}| \le C_0 \frac{\alpha n}{(d-1)^i} i^{.99} \le C_1 n 2^{-i}$$

for all $0 \le i \le .9 \log_{d-1}(n)$.

Then, for sufficiently large n, we have $2^R \leq 2^{1.1\log_{d-1}(n)} = n^{1.1\log_{d-1}(2)} \leq n^{0.7}$ for $d \geq 4$, so $n2^{-R} \geq n^{0.3}$. Additionally, we have

$$\frac{\alpha n}{(d-1)^{.9\log_{d-1}(n)}}(\log_{d-1}(n)/2)^{.99} = O(n^{0.2}),$$

so we can take $C_2 = \Theta(1)$ to be large enough such that for all $.9\log_{d-1}(n) \leq i \leq R$,

$$|B_i^{(t)}| \leq |B_{.9\log_{d-1}(n)}^{(t)}| \leq \frac{\alpha n}{(d-1)^{.9\log_{d-1}(n)}}(.9\log_{d-1}(n))^{.99} \leq C_2 n 2^{-R} \leq C_2 n 2^{-i}.$$

And as $B_i^{(t)} = \emptyset$ for all $i \geq R$, $C = \max(C_1, C_2)$ works for all $i \in \mathbb{N}$. ☐

This completes the proof of Theorem 1.1 for all $d \geq 6$.

5.2 Improved Expansion Properties for Smaller d

Just to get the $d = 4$ and $d = 5$ cases of Theorem 1.1 as well (and to improve the exponent of the logarithm for $d = 3$), we need a more careful analysis. In this section, we will prove the following stronger version of Lemma 5.1:

Lemma 5.8. *There is a $\tau = \Theta(1)$ such that the following holds. Let $a_3 = 8.1$, $a_4 = 15$, $a_5 = 24$, and $a_d = (d-1)e^{d-1}$ for all $d \geq 6$. For any $1 \leq s \leq \tau n$, define*

$$x_s = \begin{cases} 0 & \text{if } |S| \leq \log(n)/(2d) \\ \frac{\log_d(a_d)}{\log_d(|X|/|S|)-1} & \text{if } \log(n)/(2d) \leq |S| \leq \tau n \end{cases}$$

With high probability, we have that for all $S \subseteq X$ with $|S| \leq \tau n$ that

$$|N(S)| \geq (d-1-x_{|S|})|S|.$$

Then, note that the exact value of a_d is never used in the proof of Lemma 5.2 and Lemma 5.4 in [12] and we can assume $|S| \leq \tau n$ by Lemma 2.1. Therefore, the proof in [12] goes through to give insertion time $O(\log^{1+b_d}(n))$ for all $d \geq 3$. Let $b_d = \frac{\log(a_d)}{(d-1)\log(d-1)}$. When we have $b_d < .98$, our proof in Subsection 5.1 goes through to prove Lemma 3.3 and finish Theorem 1.1. We get $b_d < .98$ for $d \geq 4$, while we only get $b_3 \leq 1.509$.

To prove Lemma 5.8, need a more accurate count on the number of ways that $|N(S)|$ could take on a given value, and thus we use Stirling numbers of the second kind, $\left\{ a \atop b \right\}$, where $b!\left\{ a \atop b \right\}$ counts the number of labelled surjections from $[a]$ into $[b]$. We use the following approximation for Stirling numbers of the second kind due to Moser and Wyman:

Lemma 5.9 (Equation (5.1) of [22])**.** *If $a = bg$ for some constant $g > 1$, we have that*

$$b!\left\{ a \atop b \right\} = \left(1 \pm O\left(\frac{1}{a}\right)\right)\frac{a!(e^r-1)^b}{2r^a\sqrt{hb}}$$

where r is the solution to $\frac{r}{1-e^{-r}} = g$ and $h = \frac{\pi r e^r(e^r-1-r)}{2(e^r-1)^2}$.

Let x_s be as in Lemma 5.8. For $S \subseteq X$ with $|S| \leq \tau n$, we say that S is a failing set if $N(S) < (d-1-x_s)s$.

Lemma 5.10. *Let $v_3 = 7.266$, $v_4 = 14.986$, $v_5 = 25.5$, and $v_d = (d-1)e^{d-1}$ for all $d \geq 6$. There exists some $\tau, \zeta = \Theta(1)$ such that for all $S \subseteq X$ with $\log\log(n) \leq |S| \leq \tau n$, for sufficiently large n*

$$\mathbb{P}(S \textit{ is a failing set}) \leq \zeta m^{-x_s s - s} s^{x_s s + s} (v_d)^s.$$

Proof. Fix $S \subseteq X$ with $\log\log(n) \leq |S| \leq \tau n$. Let $s = |S|$ and let $\sigma = \lfloor (d-1-x_s)s \rfloor$. We will assume that $d \leq 5$, as for $d \geq 6$ this follows from the proof of Lemma 5.1 (Proposition 2.4 of [12]). Then

$$\mathbb{P}\left(|N(S)| < (d-1-x_{|S|})|S|\right) = \sum_{i=0}^{\sigma} \mathbb{P}\left(\exists R \in \binom{Y}{i} \text{ s.t. } N(S) = R\right) = m^{-ds} \sum_{i=0}^{\sigma} \binom{m}{i} i! \left\{ {ds \atop i} \right\}$$

We will now show that the sum above is dominated by the $i = \sigma$ term. Let $a, b \in \mathbb{N}$ with $a \geq b+1$ and let $\Theta(a,b)$ be the set of partitions of $[a]$ into b unlabelled parts (we have $|\Theta(a,b)| = \left\{ {a \atop b} \right\}$). We consider pairs (θ_1, θ_2) where the $\theta_1 \in \Theta(a,b), \theta_2 \in \Theta(a, b+1)$ and the second partition is a refinement of the first, that is, is obtained from the first by splitting a set. Now, for $\theta_1 \in \Theta(a,b)$, let $d_L(\theta_1)$ denote the number of times a partition θ occurs first in such a pair and, analogously for $\theta_2 \in \Theta(a, b+1)$, let $d_R(\theta_2)$ denote the number of times a partition occurs second in such a pair. Then

$$d_L(\theta_1) \geq \min\left\{ \sum_j 2^{x_j} - 2 : x_1 + \cdots x_b = a \right\} \geq b(2^{a/b} - 2).$$
$$d_R(\theta_2) \leq \binom{b+1}{2}$$

Because $\sum_{\theta_1 \in \Theta(a,b)} d_L(\theta_1) = \sum_{\theta_2 \in \Theta(a,b+1)} d_R(\theta_2)$, we have

$$b(2^{a/b} - 2) \left\{ {a \atop b} \right\} \leq \binom{b+1}{2} \left\{ {a \atop b+1} \right\}.$$

Let $u_i = \binom{m}{i} i! \left\{ {ds \atop i} \right\}$ for some $0 \leq i \leq \sigma$. Then we have

$$\begin{aligned} \frac{u_{i+1}}{u_i} &\geq \frac{m-i}{i+1} \cdot (i+1) \cdot \frac{i(2^{ds/i} - 2)}{\binom{i+1}{2}} = \frac{2(m-i)(2^{ds/i} - 2)}{i+1} \\ &\geq \frac{2(m - (d-1)\tau cm)(2^{d/(d-1)} - 2)}{(d-1)\tau cm} && \text{as } i \leq (d-1)s \leq (d-1)\tau cm \\ &\geq \frac{4(1 - (d-1)\tau c)(2^{1/(d-1)} - 1)}{(d-1)\tau c} > 1 && \text{if } \tau < 1/(8c) \text{ and } d \leq 5. \end{aligned}$$

Thus $\sum_{i=0}^{\sigma} u_i \leq \zeta u_s$ for some constant $\zeta > 0$. So,

$$\mathbb{P}\left(|N(S)| < (d-1-x_{|S|})|S|\right) \leq \zeta m^{-ds} \binom{m}{\sigma} \sigma! \left\{ {ds \atop \sigma} \right\}$$

Then $\frac{d}{d-1} - 0.00001 < \frac{ds}{\sigma} < \frac{d}{d-1}$ for sufficiently small τ (to get $x_{\tau n} < 0.00001$). We now use Lemma 5.9.

The numbers below are shown for $d = 5$. The proofs of $d = 3$ and $d = 4$ go through in the same way. For $d = 5$, we get $r \approx 0.46421$ and $h \approx 0.42061$, so

$$\sigma!\left\{ {5s \atop \sigma} \right\} \leq \frac{(5s)!(0.5908)^{\sigma}}{2(0.4642)^{5s}\sqrt{.4206t}} \leq \frac{(1.84s)^{5s}(0.5908)^{4s}}{(0.4642)^{5s}} \leq s^{5s}(119.22)^{s},$$

and

$$\begin{aligned} m^{-ds}\binom{m}{\sigma}\sigma!\left\{ {ds \atop \sigma} \right\} &\leq m^{-5s}\left(\frac{em}{\sigma}\right)^{\sigma} s^{5s}(119.22)^{s} \\ &\leq m^{-x_s s - s} e^{4\sigma}(3.999s)^{-\sigma} s^{5s}(119.23)^{s} \\ &\leq m^{-x_s s - s} s^{x_s s + s}\left[119.23(e^{4})(3.999^{-3.999})\right]^{s} \\ &\leq m^{-x_s s - s} s^{x_s s + s}\left[25.5\right]^{s}. \end{aligned}$$

The following table shows what some of the intermediate numbers are for $3 \leq d \leq 5$:

	$d = 3$	$d = 4$	$d = 5$
r	.87422	.60586	.46421
$e^r - 1$	1.397	.8329	.5908
$(d/e)^d(e^r - 1)^{d-1}r^{-d}$	3.9266	20.102	119.22
v_d (previous # times $e^{d-1}(d-1)^{-(d-1)}$)	7.266	14.986	25.5

□

Recall that for $S \subseteq X$ with $|S| \leq \tau n$, S is a failing set if $N(S) < (d - 1 - x_s)s$. Now, we say that S is a minimal failing set if S is a failing set but for every $R \subsetneq S$, R is not a failing set.

Lemma 5.11. *There exists some $\tau = \Theta(1)$ such that for all $S \subseteq X$ with $\log\log(n) \leq |S| \leq \tau n$, for sufficiently large n*

$$\mathbb{P}(S \textit{ is a minimal failing set}) \leq \mathbb{P}(S \textit{ is a failing set})(q_d)^{|S|}$$

for some q_d where $q_3 \leq .446$, $q_4 \leq .376$, $q_5 \leq .347$, and $q_d \leq \frac{1}{e}$ for all $d \geq 6$.

Proof. We create the ds random hashes from S in two steps: first, we cast ds balls into m bins. Then, we randomly assign the ds balls to the ds elements of $S \times [d]$. Note that whether or not S is a failing set only depends on the first step. Therefore, we just need to show that if the first step creates a failing set, the second step will only create a minimal failing set with probability $\leq (q_d)^s$.

Let $x \in S$. If $S \setminus \{x\}$ is not a failing set but S is, we have that

$$|N(S\setminus\{x\})| \geq (d-1-x_{s-1})(s-1) \geq (d-1-x_s)(s-1) > |N(S)|-(d-1-x_s) \geq |N(S)|-(d-1). \tag{3}$$

In particular, this means that for S to possibly be a minimal failing set, we must have $|N(S)| \geq |N(S \setminus \{x\})| \geq (d - 1 - x_s)(s - 1)$. So after casting the ds balls into the m bins, and thus determining $|N(S)|$, we can assume that we have $(d-1-x_s)(s-1) \leq |N(S)| < (d-1-x_s)s$, that is, it suffices to show that in this case, the probability of S being a minimal failing set is $\leq (q_d)^s$, as in other cases S is not a minimal failing set.

Let $A \subseteq [ds]$ be the set of balls that ended up in a bin with another ball.

$$|A| \leq 2(ds - |N(S)|) \leq 2(ds - (d-1-x_s)(s-1)) = 2(1+x_s)(s-1) + 2d \leq 2.001s.$$

Now, we go about assigning A to a random subset A' of $S \times [d]$. If there is some $x \in S$ for which $|(x \times [d]) \cap A'| < 2$, then

$$|N(S \setminus \{x\})| \leq N(S) - d + |(x \times [d]) \cap A'| \leq N(S) - d + 1$$

which is a contradiction to Equation (3), that is, $S \setminus \{x\}$ becomes a failing set.

Therefore, the probability that S is a minimal failing set is at most the probability that $|(x \times [d]) \cap A'| \geq 2$ for every $x \in S$. Clearly, this is impossible (probability 0) if $|A| < 2|S|$, so it suffices to show that for every $2s \leq |A| \leq 2.001s$, the probability of A' satisfying $|(x \times [d]) \cap A'| \geq 2$ for every $x \in S$, conditioned on $|A|$, is at most $(q_d)^s$.

Assume that we have thrown the balls and thus fixed A. The total number of equally likely possibilities for A' is, for $d \leq 4$

$$\binom{ds}{|A|} \geq \binom{ds}{2.001s} \geq 2^{dsH(2.001/d)}(ds+1)^{-1}$$

(where $H(p) = -p\log_2(p) - (1-p)\log_2(1-p)$) or, for $d \geq 5$,

$$\binom{ds}{|A|} \geq \binom{ds}{2s} \geq 2^{dsH(2/d)}(ds+1)^{-1}$$

The total number of possibilities for A' that satisfy the condition $|(x \times [d]) \cap A'| \geq 2$ for every $x \in S$ is at most

$$\binom{d}{2}^s \binom{ds}{|A|-2s} \leq \left(\frac{d(d-1)}{2}\right)^s \binom{ds}{.001s} \leq \left[\frac{d(d-1)(1000ed)^{.001}}{2}\right]^s$$

Thus, for $d \leq 4$,

$$\frac{\mathbb{P}(S \text{ min. failing set})}{\mathbb{P}(S \text{ failing set})} \leq \left[\frac{d(d-1)(1000ed)^{.001}(ds+1)^{1/s}}{2^{1+dH(2.001/d)}}\right]^s$$

and for $d \geq 5$,

$$\frac{\mathbb{P}(S \text{ min. failing set})}{\mathbb{P}(S \text{ failing set})} \leq \left[\frac{d(d-1)(1000ed)^{.001}(ds+1)^{1/s}}{2^{1+dH(2/d)}}\right]^s.$$

This expression is less than q_d for all $3 \leq d \leq 10$ and sufficiently large n. If we ignore the $(1000ed)^{.001}(ds+1)^{1/s}$ term in the expression (which can be removed in the limit by making τ depend on d), the limit of this expression as $d \to \infty$ is $\left(\frac{2}{e^2}\right)^s \approx 0.271^s$. $\square$

Now, we have all the ingredients we need to prove our improved expansion lemma.

Proof of Lemma 5.8. For Lemma 5.8 to fail, there must be some $S \subseteq X$ with $|S| \leq \tau n$ such that S is a minimal failing set. Then

$$\begin{aligned}
\mathbb{P}(\text{Lemma 5.8 fails}) &\leq \sum_{s=1}^{\tau n} \mathbb{P}(\exists\, S \in \binom{X}{s} \text{ s.t. } S \text{ is a minimal failing set}) \\
&\leq \sum_{s=1}^{\log(n)/(2d)} \mathbb{P}(\exists\, S \in \binom{X}{s} \text{ s.t. } S \text{ is a failing set}) \\
&\qquad + \sum_{s=\log(n)/(2d)}^{\tau n} \mathbb{P}(\exists\, S \in \binom{X}{s} \text{ s.t. } S \text{ is a minimal failing set}) \\
&\leq \sum_{s=1}^{\log(n)/(2d)} \frac{ds}{n} \left(c_d^*(d-1)e^d\right)^s \\
&\qquad + \sum_{s=\log(n)/(2d)}^{\tau n} \mathbb{P}(\exists\, S \in \binom{X}{s} \text{ s.t. } S \text{ is a minimal failing set}) \\
&\qquad\qquad \text{(by the proof of Proposition 2.4, [12])} \\
&\leq O(n^{-1/5}) + \sum_{s=\log(n)/(2d)}^{\tau n} \binom{n}{s} (q_d)^s \mathbb{P}(S \text{ is a failing set}) \\
&\qquad\qquad \text{(by Lemma 5.11)} \\
&\leq O(n^{-1/5}) + \sum_{s=\log(n)/(2d)}^{\tau n} \binom{cm}{s} (q_d)^s \zeta m^{-x_s s - s} s^{s + x_s s} (v_d)^s \\
&\qquad\qquad \text{(by Lemma 5.10)} \\
&\leq O(n^{-1/5}) + \zeta \sum_{s=\log(n)/(2d)}^{\tau n} (c_d^* e)^s m^s s^{-s} m^{-x_s s - s} s^{s + x_s s} (q_d v_d)^s \\
&\leq O(n^{-1/5}) + \zeta \sum_{s=\log(n)/(2d)}^{\tau n} \left[(s/m)^{x_s} c_d^* q_d v_d e\right]^s \\
&\leq O(n^{-1/5}) + \zeta \sum_{s=\log(n)/(2d)}^{\tau n} 0.999^s \\
&= o(n^{-\eta}) \qquad \text{for some small } \eta = \Theta(1)
\end{aligned}$$

when we take $x_s = \log_{m/s}(a_d) = \frac{\log_d(a_d)}{\log_d(m/s)} \leq \frac{\log_d(a_d)}{\log_d(|X|/|S|)-1}$, noting that we have set $a_d > c_d^* q_d v_d e/0.999$. □

References

[1] Martin Aumüller, Martin Dietzfelbinger, and Philipp Woelfel. Explicit and efficient hash families suffice for cuckoo hashing with a stash. *Algorithmica*, 70:428–456, 2014.

[2] Julie Anne Cain, Peter Sanders, and Nick Wormald. The random graph threshold for k-orientiability and a fast algorithm for optimal multiple-choice allocation. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, page 469–476, 2007.

[3] Jeffrey S. Cohen and Daniel M. Kane. Bounds on the independence required for cuckoo hashing. 2009.

[4] Luc Devroye and Pat Morin. Cuckoo hashing: Further analysis. *Information Processing Letters*, 86(4):215–219, 2003.

[5] Martin Dietzfelbinger, Andreas Goerdt, Michael Mitzenmacher, Andrea Montanari, Rasmus Pagh, and Michael Rink. Tight thresholds for cuckoo hashing via xorsat. *Proceedings of the 37th International Colloquium Conference on Automata, Languages and Programming (ICALP)*, pages 213–225, 2010.

[6] Martin Dietzfelbinger and Christoph Weidling. Balanced allocation and dictionaries with tightly packed constant size bins. *Theoretical Computer Science*, 380(1):47–68, 2007.

[7] David Eppstein, Michael T. Goodrich, Michael Mitzenmacher, and Paweł Pszona. Wear minimization for cuckoo hashing: How not to throw a lot of eggs into one basket. *Proceedings of the International Symposium on Experimental Algorithms (SEA)*, pages 162–173, 2014.

[8] Daniel Fernholz and Vijaya Ramachandran. The k-orientability thresholds for $G_{n,p}$. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 459–468, 2007.

[9] Dimitris Fotakis, Rasmus Pagh, Peter Sanders, and Paul G. Spirakis. Space efficient hash tables with worst case constant access time. *Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, page 271–282, 2003.

[10] Nikolaos Fountoulakis, Megha Khosla, and Konstantinos Panagiotou. The multiple-orientability thresholds for random hypergraphs. *Proceedings of the 2017 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1222–1236, 2011.

[11] Nikolaos Fountoulakis and Konstantinos Panagiotou. Orientability of random hypergraphs and the power of multiple choices. *Proceedings of the 37th International Colloquium Conference on Automata, Languages and Programming (ICALP)*, pages 348–359, 2010.

[12] Nikolaos Fountoulakis, Konstantinos Panagiotou, and Angelika Steger. On the insertion time of cuckoo hashing. *SIAM Journal on Computing*, 42(6):2156–2181, 2013.

[13] Alan Frieze and Tony Johansson. On the insertion time of random walk cuckoo hashing. *Proceedings of the 2017 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1497–1502, 2017.

[14] Alan Frieze, Páll Melsted, and Michael Mitzenmacher. An analysis of random-walk cuckoo hashing. *Proceedings of the 2009 International Conference on Randomization and Computation (RANDOM)*, 2009.

[15] Alan Frieze and Páll Melsted. Maximum matchings in random bipartite graphs and the space utilization of cuckoo hash tables. *Random Structures & Algorithms*, 41(3):334–364, 2012.

[16] Alan Frieze and Samantha Petti. Balanced allocation through random walk. *Information Processing Letters*, 131:39–43, 2018.

[17] Pu Gao and Nicholas C. Wormald. Load balancing and orientability thresholds for random hypergraphs. *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC)*, pages 97–104, 2010.

[18] Megha Khosla and Avishek Anand. A faster algorithm for cuckoo insertion and bipartite matching in large graphs. *Algorithmica*, 81(9):3707–3724, 2019.

[19] Adam Kirsch, Michael Mitzenmacher, and Udi Wieder. More robust hashing: Cuckoo hashing with a stash. *SIAM Journal on Computing*, 39(4):1543–1561, 2009.

[20] Brice Minaud and Charalampos Papamanthou. Generalized cuckoo hashing with a stash, revisited. *Information Processing Letters*, 181(106356), 2023.

[21] Michael Mitzenmacher. Some open questions related to cuckoo hashing. *Proceedings of the 17th Annual European Symposium on Algorithms (ESA)*, pages 1–10, 2009.

[22] L. Moser and M. Wyman. Stirling numbers of the second kind. *Duke Mathematical Journal*, 25(1):29–43, 1958.

[23] Rasmus Pagh and Flemming Friche Rodler. Cuckoo hashing. *Proceedings of the 9th Annual European Symposium on Algorithms (ESA)*, pages 121–133, 2001.

[24] Yuanyuan Sun, Yu Hua, Dan Feng, Ling Yang, Pengfei Zuo, Shunde Cao, and Yuncheng Guo. A collision-mitigation cuckoo hashing scheme for large-scale storage systems. *IEEE Transactions on Parallel and Distributed Systems*, 28(3):619–632, 2017.

[25] Stefan Walzer. Insertion time of random walk cuckoo hashing below the peeling threshold. *Embedded Systems and Applications*, 2022.

[26] Kevin Yeo. Cuckoo hashing in cryptography: Optimal parameters, robustness and applications. *43rd Annual International Cryptology Conference (CRYPTO)*, page 197–230, 2023.